IN THE DRAWINGS

The attached sheets of drawings include changes to Figures 2 and 3. These sheets replace the original sheets, including Figures 2 and 3.

REMARKS

Reconsideration of this application as amended is respectfully requested.

In the Office Action dated October 29, 2007, claims 1-30 are pending. Claims 1-30 stand rejected. In this response, claims 4, 8, 9-14, 17, and 23-29 have been amended. Claim 30 has been canceled without prejudice. No new claims have been added. Thus, claims 1-29 remain pending. Support for the amendments can be found throughout the specifications as filed. No new matter has been added. Applicants reserve all rights with respect to the applicability of the Doctrine of Equivalents.

Objection to the Specification

The specification is objected to because of the presence of possible minor errors. In view of the foregoing amendments, applicants respectfully submit that the errors in the specification applicants are aware of have been corrected.

Objections to the Drawings

Figures **1 and 2**

Figures 1 and 2 are objected to for failing to be designated as "Prior Arts". Office Action asserts "Figured 1 and 2 should be designated by a legend such as --Prior Art-- because only that which is old is illustrated" (Office Action, Page 2). Applicants respectfully disagree.

Office Action states "The figures show prior art systems usable with or by the claimed system, but do not show any structure representing an aspect of the claimed system" (Office Action, Page 2). It appears that the Office Action asserts the alleged admitted prior art without interpreting the drawings in view of the specifications. On the contrary, these drawings illustrate embodiments of the invention that together with the description serve to explain the principles of the invention (Specification, BRIEF DESCRIPTION OF THE DRAWINGS, [0005]).

Specifically, for example, Figure 1 contains an exemplary system performing the processes shown in Figures 5-8 (Specification, [0029]). A software architecture of a multithreading system executing one or more generated helper threads using parallelization analysis is included as illustrated in Figures 5-8 ([0011]-[0014]). Additionally, Figure 2 shows an embodiment of a computing system capable of performing disclosed techniques in the instant application, including, for example, speculative threads (Specification, [0038][0039]). Apparently, the instant application discloses techniques including the claimed invention. Thus, Figures 1 and 2 are clearly part of the invention as claimed.

Therefore, Figures 1 and 2, in view of the specification as filed, clearly illustrate embodiments of the invention as claimed. Withdrawal of the objection is respectfully requested.

Figures **2 and 3**

Figures 2 and 3 are objected to as failing to comply with 37 CFR 1.84(p)(5). In view of the foregoing amendments, applicants respectfully submit that Figure 2 and 3, as amended, are in compliance of the requirements of 37 CFR 1.84(p)(5). Withdrawal of the objection is respectfully requested.

Objections to the Claims

Claim **30**

Claim 30 is objected to under 37 CFR 1.75(c) as being of improper dependent form. Claim 30 has been canceled without prejudice. Withdrawal of the objection of claim 30 is respectfully requested.

Rejections under 35 U.S.C. § 101

Claims **8-14 and 23-28**

Claims 8-14 and 23-28 stand rejected under 35 U.S.C. §101 because the claimed invention is allegedly directed to non-statutory subject matter. Claims 8-14 and 23-28 have been amended. In view of the foregoing amendments, applicants respectfully submit that claims 8-14 and 23-28, as amended, are directed to statutory subject matter. Accordingly, applicants respectfully request withdrawal of the rejection of claims 8-14 and 23-28 under 35 U.S.C. § 101.

Rejections under 35 U.S.C. § 112

Claims **4 and 11**

Claims 4 and 11 stand rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter. In view of the foregoing amendments, applicants respectfully submit that claims 4 and 11, as amended, are definite in compliance with the requirements of 35 U.S.C. §112, second paragraph. Accordingly, applicants respectfully request withdrawal of the rejection of claims 4 and 11 under 35 U.S.C. § 112.

Rejections under 35 U.S.C. § 102(b)

Claims **1-2, 8-9 and 15-16**

Claims 1-2, 8-9 and 15-16 stand rejected under 35 U.S.C. §102(b) as being anticipated by "Tolerating Memory Latency through Software-Controlled Pre-Execution in Simultaneous Multithreading Processors" by Luk (hereinafter "Luk"). However, applicants respectfully submit that applicants' claims 1-2, 8-9 and 15-16, as amended, are not anticipated by the cited reference.

Specifically, for example, independent claim 1 includes the limitations:

> "identifying a region of a main thread that likely has one or more delinquent loads, the one or more delinquent loads representing loads which likely suffer cache misses during an execution of the main thread;
> analyzing the region for one or more helper threads with respect to the main thread; and
> generating code for the one or more helper threads, the one or more helper threads being speculatively executed in parallel with the main thread to perform one or more tasks for the region of the main thread"
> (emphasis added)

Applicants' claim 1 includes the limitation of identifying a region of a main thread to analyze the region for one or more helper threads. It is respectfully submitted that Luk fails to disclose or suggest the noted limitations.

Rather, Luk provides software-controlled pre-execution schemes to help programmers writing memory-friendly programs using pre-execution threads and for compiler to insert pre-execution (Luk, page 40, col. 2, par. 4 – page 41, col. 1, par . 1). Luk proposes ISA (Instruction Set Architecture) extension with three new instructions to spawn a pre-execution thread at a particular PC (Program Counter) and stops when certain number of instructions have been executed in the pre-execution (Luk, page 44, col. 1, pars. 4-5, col. 1, par. 1).

According to Luk, a compiler inserts pre-execution by determining which references could benefit from the pre-execution and calculating the pre-execution distance (Luk, page 44, col. 2, par. 4). In addition, Luk's pre-execution thread stops either at a pre-determined PC (program counter) or when a sufficient number of instructions have been pre-executed (Luk, page 41, col. 2, par.1). Thus, Luk determines to terminate pre-execution independent of where in the main thread the pre-execution is started. However, Luk is completely silent about identifying a region of a main thread to analyze the region for one or more helper threads

The Office Actions notes that Luk includes locality analysis determining which references are likely to cause cache misses to identify a region of a main thread that likely has one or more delinquent loads (Office Action, page 6). Applicants respectfully disagree.

Specifically, Luk describes a local analysis which locates where cache misses occur in a program to determine if the misses are generated by access patterns that can potentially be pre-executed (Luk, page 51, col. 2, pars. Step1-Step2). Luk discusses using low-overhead profiling tools or cache simulation to locate cache misses (Luk, page 51, col. 2, par. Step 1). In addition, Luk uses compilers to recognize pointer-based data structures, array access, control-flow analysis and call-graph analysis (Luk, page 51, col. 2, par. Step 2). Apparently, Luk's locality analysis does not identify a region of a main thread that likely has one or more delinquent loads.

In order to anticipate a claim, each and every limitation of the claim must be taught by the cited reference. It is respectfully submitted that Luk fails to disclose the limitations set forth above. Therefore, it is respectfully submitted that independent claim 1 is not anticipated by Luk.

Independent claims 8 and 15, as amended, include similar limitations as noted above. Therefore, for at least the similar reasons as discussed above, it is respectfully submitted that claims 8 and 15, as amended, are not anticipated by Luk.

Given that claims 2, 9 and 16, as amended, depend from and include all limitations of one of independent claims 1, 8 and 15, as amended, applicants respectfully submit that claims 2, 9 and 16, as amended, are not anticipated by Luk.

Claims **17-19, 22-25 and 28-30**

Claims 17-19, 22-25 and 28-30 stand rejected under 35 U.S.C. §102(b) as being anticipated by by Luk. Claim 30 has been canceled without prejudice. However, applicants respectfully submit that applicants' claims 17-19, 22-25 and 28-29, as amended, are not anticipated by the cited reference.

Specifically, for example, independent claim 17, as amended, includes the limitations:

> "executing a main thread of an application in a multi-threading system; and
> spawning more than one helper threads from the main thread to perform one or more computations for the main thread when the main thread enters a region having one or more delinquent loads, code of the one or more helper thread being created separately from code of the main thread during a compilation of the main thread."
> (emphasis added)

Applicants' claim 17, as amended, includes the limitation of spawning one or more helper threads associated with codes generated separately from code of the main thread. It is respectfully submitted that Luk fails to disclose or suggest the noted limitations.

Rather, Luk discloses adding ISA extended instructions, namely, PreExecute_Start, PreExecute_Stop and PreExecute_Cancel for specifying when and where to start and terminate pre-execution (Luk, page 44, col. 1. par. 4). According to Luk, a pre-execution thread executes as normal (based on the main thread) (Luk, page 41, col. 2, par. 1). In addition, Luk allows only the main thread to spawn and terminate a pre-execution thread (Luk, page 44, col. 2, par. 1). Luk also states that a pre-execution thread terminates when executing the instruction PreExecute_Stop, while a main thread has no effect executing the same instruction (Luk, page 44, Figure 4). Clearly, Luk's main thread and pre-execution threads are executed based on the same code. Nowhere does Luk teach or suggest spawning one or more helper threads associated with codes generated separately from code of the main thread.

In order to anticipate a claim, each and every limitation of the claim must be taught by the cited reference. It is respectfully submitted that Luk fails to disclose the limitations set forth above. Therefore, it is respectfully submitted that independent claim 17, as amended, is not anticipated by Luk.

Independent claims 17, 23 and 29, as amended, include similar limitations as noted above. Therefore, for at least the similar reasons as discussed above, it is respectfully submitted that claims 17, 23 and 29, as amended, are not anticipated by Luk.

Given that claim 18-19, 22, 24-25 and 28, as amended, depend from and include all limitations of one of independent claims 17 and 23, as amended, applicants respectfully submit that claims 18-19, 22, 24-25 and 28, as amended, are not anticipated by Luk.

Rejections Under 35 U.S.C. 103(a)

Claims **3-4, 10-11, 5-7, 12-14, 20-21, 26-27**

Claims 3-4 and 10-11 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Luk in view of "Exploiting hardware Performance Counters with Flow and Context Sensitive Profiling" by Ammons et al. (hereinafter "Ammons"). Claims 5-7 and 12-14 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Luk in view of "Data Prefetching by Dependence Graph Precomputation" by Annavaram et al. (hereinafter "Annavaram"). Claims 20-21 and 26-27 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Luk in view of US Patent No. 7,243,267 to Kelmm et al. (hereinafter "Klemm"). Applicants hereby reserve the right to swear behind Kelmm at a later date. However, applicants respectfully submit that applicants' claims 3-4, 10-11, 5-7, 12-14, 20-21 and 26-27, as amended, are patentable over the cited references.

These claims depend from one of the above independent claims. It is respectfully submitted that none of the cited references herein, individually or in combination, disclose or suggest the limitations set forth above. Furthermore, there are neither motivations nor suggestions to combine the cited references. Therefore, for the reasons similar to those discussed above, these claims are patentable over the cited references.

CONCLUSION

In view of the foregoing, it is respectfully submitted that the applicable rejections and objections have been overcome.

Authorization is now given to charge our Deposit Account No. 02-2666 for any charges that may be due. Furthermore, if an extension is required, then applicant hereby requests such an extension.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Date: 1-10 ,2008

Vincent WenJeng Lue
Reg. No. 56,564
Vincent_Lue@bstz.com

1279 Oakmead Parkway
Sunnyvale, California 94086-4040
(408) 720-8300